Assignment 5-4　Perfect Numbers

A perfect number is a positive integer that is equal to the sum of its positive proper divisors, that is, divisors excluding the number itself. For instance, 6 has proper divisors 1, 2 and 3, and $1 + 2 + 3 = 6$, so 6 is a perfect number. The next perfect number is 28, since $1 + 2 + 4 + 7 + 14 = 28$.

You would be given several integer numbers. You have to determine if these numbers are perfect numbers. The largest perfect number in this problem will not exceed $2^{31} - 1$.

Input

The input consists of t ($30 \le t \le 40$) test cases. The first line of the input contains only positive integer t. Then t test cases follow. Each test case consists of exactly one line with a positive integer n which is less than 2^{31}.

Output

For each such integer n, you are to output a single line containing "perfect number" or "non-perfect number" depending on whether the integer n is a perfect number.

Sample Input

2
28
50

Sample Output

perfect number
non-perfect number

Requirements

You are required to write a recursive function `int sumFactors( int n, int start )` to complete the following program which solves this problem. This function returns the sum of factors of `n` in the range [`start .. n - 1`].

```
#include <iostream>
using namespace::std;

// find the sum of factors of n in the range [ start .. n - 1 ]
int sumFactors( int n, int start );

int main()
```

```
{
    int numCases;
    cin >> numCases;
    for( int i = 1; i <= numCases; i++ )
    {
        int n;
        cin >> n;

        // n is equal to the sum of factors of n in the range [ 1 .. n - 1 ]
        if( n == sumFactors( n, 1 ) )
            cout << "perfect number" << endl;
        else
            cout << "non-perfect number" << endl;
    }
}

int sumFactors( int n, int start )
{




}
```